



# UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE

United States Patent and Trademark Office

Address: COMMISSIONER FOR PATENTS

P.O. Box 1450

Alexandria, Virginia 22313-1450

www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/679,254	10/02/2003	Manuvir Das	3382-65960	4134
26119 7590 07/24/2008 KLARQUIST SPARKMAN LLP 121 S.W. SALMON STREET SUITE 1600 PORTLAND, OR 97204				
EXAMINER				
KISS, ERIC B				
ART UNIT		PAPER NUMBER		
2192				
MAIL DATE		DELIVERY MODE		
07/24/2008		PAPER		

**Please find below and/or attached an Office communication concerning this application or proceeding.**

The time period for reply, if any, is set in the attached communication.

### Office Action Summary

**Application No.**

10/679,254

**Applicant(s)**

DAS ET AL.

**Examiner**

Eric B. Kiss

**Art Unit**

2192

**-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --**  
**Period for Reply**

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

**Status**

- 1) ☒ Responsive to communication(s) filed on 30 April 2008.
- 2a) ☐ This action is **FINAL**. 2b) ☒ This action is non-final.
- 3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

**Disposition of Claims**

- 4) ☒ Claim(s) 7, 14, 15, 21, 24, 27-34, 41-46 and 53-60 is/are pending in the application.
- 4a) Of the above claim(s) \_\_\_\_\_ is/are withdrawn from consideration.
- 5) ☐ Claim(s) \_\_\_\_\_ is/are allowed.
- 6) ☒ Claim(s) 7, 14, 15, 21, 24, 27-34, 41-46 and 53-60 is/are rejected.
- 7) ☐ Claim(s) \_\_\_\_\_ is/are objected to.
- 8) ☐ Claim(s) \_\_\_\_\_ are subject to restriction and/or election requirement.

**Application Papers**

- 9) ☐ The specification is objected to by the Examiner.
- 10) ☐ The drawing(s) filed on \_\_\_\_\_ is/are: a) ☐ accepted or b) ☐ objected to by the Examiner.  
Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).  
Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
- 11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

**Priority under 35 U.S.C. § 119**

- 12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
- a) ☐ All b) ☐ Some \* c) ☐ None of:
1. ☐ Certified copies of the priority documents have been received.
  2. ☐ Certified copies of the priority documents have been received in Application No. \_\_\_\_\_.
  3. ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

\* See the attached detailed Office action for a list of the certified copies not received.

**Attachment(s)**

- 1) ☐ Notice of References Cited (PTO-892)
- 2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
- 3) ☐ Information Disclosure Statement(s) (PTO/SB/08)  
Paper No(s)/Mail Date \_\_\_\_\_
- 4) ☐ Interview Summary (PTO-413)  
Paper No(s)/Mail Date \_\_\_\_\_
- 5) ☐ Notice of Informal Patent Application
- 6) ☐ Other: \_\_\_\_\_

## **DETAILED ACTION**

### ***Continued Examination Under 37 CFR 1.114***

1. A request for continued examination under 37 CFR 1.114, including the fee set forth in 37 CFR 1.17(c), was filed in this application after final rejection. Since this application is eligible for continued examination under 37 CFR 1.114, and the fee set forth in 37 CFR 1.17(c) has been timely paid, the finality of the previous Office action has been withdrawn pursuant to 37 CFR 1.114. Applicant's submission filed on April 30, 2008, has been entered. Claims 7, 14, 15, 21, 24, 27-34, 41-46, and 53-60 are pending.

### ***Response to Arguments***

2. Applicant's arguments filed April 30, 2008, have been fully considered but they are not persuasive.

Regarding claim 7, the examiner asserts that the Splint manual discloses, "wherein the annotations on the first pointer are placed in an argument list to a function call that uses the first pointer as a parameter." In the examples on p. 49, the annotations occur between the function name and the semicolon indicating the end of the function call, and thus may be considered part of the argument list.

Regarding claim 24, the examiner asserts that the Splint manual discloses, "wherein use of the keyword associates a pre-determined set of usability properties with a value type." See, e.g., section 4, beginning on p. 19, describing annotations associated with types and the corresponding checking associated with the various type-specific annotations.

Regarding claim 30, the examiner maintains that the alt annotation (e.g., p. 57 of the Splint manual) overrides the type checking by adding alternative types, which incorporates the

type checking associated with the newly added types (in the examples on p. 57, overriding the type checking associated with integers with that associated with integers OR floating point values).

Regarding claim 34, the examiner asserts that the Splint manual likewise details handling of aliasing through specific annotations (as described, e.g., in section 6.1, beginning on p. 31 of the Splint manual).

Regarding claim 53, the examiner maintains that the Splint manual discloses, in the annotation context through the use of the alt annotation, overriding the type checking by adding alternative types, which incorporates the type checking associated with the newly added types (in the examples on p. 57, overriding the type checking associated with integers with that associated with integers OR floating point values).

***Claim Rejections - 35 USC § 102***

3. The text of those sections of Title 35, U.S. Code not included in this action can be found in a prior Office action.
4. Claims 7, 14, 15, 21, 24, 27-34, 41-46, and 53-60 are rejected under 35 U.S.C. 102(a) as being anticipated by Evans et al., “Splint Manual, Version 3.1.1-1,” June 5, 2003 (prior art of record; hereinafter “[Splint]”).

As per claim 7, [Splint] discloses:

inserting one or more in-line code annotations at one or more annotation targets in source code (see, e.g., p. 13, describing annotations);

wherein the one or more annotations comprise at least one annotation on a first pointer to a buffer, wherein the at least one annotation comprises a property that indicates a characteristic

of the buffer, wherein the property that indicates the characteristic of the buffer takes a size argument, and wherein the size argument comprises a location of a second pointer associated with the buffer (see, e.g., the sample buffer annotations on p. 49 (in the first example, the size parameters are specified through maxSet (the highest address (i.e., a pointer) that can be safely used as an lvalue) and maxRead (the highest index of a buffer (i.e., a pointer specified as an offset) that can be safely used as an rvalue), and in the second example, an additional size argument, "size\_t n" is provided and this size indicates the location of the end pointer of the buffer (i.e., as an offset)); and wherein the annotations on the first pointer are placed in an argument list to a function call that uses the first pointer as a parameter (in the examples on p. 49, the annotations occur between the function name and the semicolon indicating the end of the function call, and thus may be considered part of the argument list).

As per claim 14, [Splint] further discloses the characteristic is a readable extent of the buffer (see, e.g., section 9.1, describing the maxRead property).

Regarding claim 44, [Splint] further discloses using the location of the second pointer associated with the buffer to determine the readable extent of the buffer (see, e.g., section 9.1, describing the maxRead property).

As per claim 15, [Splint] further discloses the characteristic is a writable extent of the buffer (see, e.g., section 9.1, describing the maxSet property).

Regarding claim 45, [Splint] further discloses using the location of the second pointer associated with the buffer to determine the writable extent of the buffer (see, e.g., section 9.1, describing the maxSet property).

As per claim 21, [Splint] further discloses the one or more code annotations include an annotation prefix (see, e.g., section 9.1, describing the maxRead property).

Regarding claim 41, [Splint] further discloses using the location of the second pointer associated with the buffer to determine the size of the buffer (see, e.g., section 9.2, describing annotating buffer sizes).

Regarding claim 42, [Splint] further discloses the second pointer associated with the buffer being an end pointer for the buffer (maxSet is the location of the writable extent of the buffer, i.e., the end pointer).

Regarding claim 43, [Splint] further discloses the second pointer associated with the buffer being an internal pointer for the buffer (see, e.g., section 9.1 (maxRead is the location of a null character within a character array buffer)).

As per claim 24, [Splint] discloses inserting an annotation at a first value having a first value type in the computer program code (see, e.g., p. 13, describing annotations);

wherein the annotation comprises a first instance of a keyword indicating that the first value satisfies all usability properties necessary to allow a first function to rely on the first value, wherein other instances of the keyword identical to the first instance are operable to indicate that other values having different respective value types satisfy all usability properties necessary to allow functions to rely on the respective other values, wherein use of the keyword associates a pre-determined set of usability properties with a value type (see, e.g., section 4, beginning on p. 19, describing annotations associated with types), and wherein the usability properties depend on the value type (see, e.g., p. 13, describing annotations; section 7, beginning on p. 35, describing annotations for function parameters; section 4, describing type checking; section 7.5, describing

requires and ensures clauses, where the keywords “requires” and “ensures” are usable in multiple type contexts).

As per claim 27, [Splint] further discloses the first value type is scalar, void, pointer, user-defined type, or struct (see, e.g., section 4, describing type checking).

As per claim 28, [Splint] further discloses the first value is a reference parameter (see, e.g., section 9, beginning on p. 48, describing buffers accessed by pointers).

As per claim 29, [Splint] further discloses the first value is a pointer, wherein an object pointed to by the pointer has one or more readable elements, the one or more readable elements of the object each having usability properties sufficient to allow the first function to rely on the one or more readable elements (see, e.g., section 9, beginning on p. 48, describing buffers accessed by pointers).

Regarding claim 46, [Splint] further discloses the annotation further comprising an except qualifier (see, e.g., section 11.2.1, describing side effect free parameters).

As per claim 30, [Splint] discloses inserting an annotation having an argument in the computer program code, wherein the annotation annotates a value having a first declared value type with a first set of annotation-specific usability properties (see, e.g., p. 13, describing annotations; section 7, beginning on p. 35, describing annotations for function parameters; section 4, describing type checking);

wherein the annotation overrides the first set of annotation-specific usability properties of the first declared value type and indicates that the value has annotation-specific usability properties that depend on annotation-specific properties of a second value type denoted by the argument of the annotation (see, e.g., pp. 24 and 57, describing the alt annotation; the alt

annotation overrides the type checking by adding alternative types, which incorporates the type checking associated with the newly added types).

As per claim 31, [Splint] further discloses the first value type is a legacy value type (see, e.g., section 4.2, describing pre-ISO99 C Boolean representation).

As per claim 32, [Splint] further discloses the first value type is void \* and wherein the second value type has a null-termination characteristic (see, e.g., p. 49, showing the annotated strncpy as returning void or alternatively a character buffer).

As per claim 33, [Splint] further discloses the first value type is char \* and wherein the second value type has a null-termination characteristic (see, e.g., section 9.2).

As per claim 34, [Splint] discloses adding an annotation to a pointer in the computer program code, wherein the annotation describes transferring buffer properties from a second pointer to the pointer; and including a location parameter with the annotation, wherein the location parameter describes the logical buffer pointed to by the pointer (see, e.g., section 6, beginning on p. 31).

Regarding claim 53, [Splint] discloses:

reading at least one annotation having an argument from the annotated computer program code, wherein the at least one annotation annotates a value having a first declared value type with a first set of usability properties used only in an annotation context, and wherein the annotation overrides the first set of usability properties used only in the annotation context of the first declared value type and indicates a second set of usability properties used only in the annotation context for the value that depend on the second value type denoted by the argument of the annotation such that the second set of usability properties are used in the context of a second



annotation rather than the first set of usability properties (see, e.g., pp. 24 and 57, describing the alt annotation; the alt annotation overrides the type checking by adding alternative types, which incorporates the type checking associated with the newly added types);

processing the annotated computer program code based at least in part on the second set of usability properties for the value (see, e.g., pp. 24 and 57, describing uses of the alt annotation); and

outputting a result of the processing (the outputting of results is disclosed throughout the Splint manual (see, e.g., pp. 11-12), and various descriptions of warning message handling is found on pp. 24 and 57).

Regarding claim 54, [Splint] further discloses the second value type is a null-terminated string type (see, e.g., p. 49, showing the annotated strncpy as returning void or alternatively a character buffer).

Regarding claim 55, [Splint] further discloses the processing comprises determining whether the value satisfies the second set of usability properties (see, e.g., pp. 24 and 57, describing the functionality of the alt annotation).

Regarding claim 56, [Splint] further discloses if the second value type is a pointer, the usability properties necessary to allow the first function to rely on the first value comprising the pointer pointing to a buffer with at least one readable element (see, e.g., p. 48 (MaxRead(ptr)>=0)).

Regarding claim 57, [Splint] further discloses if the first value type is a scalar, the usability properties necessary to allow the first function to rely on the first value comprising the scalar being initialized (see, e.g., section 3, beginning on p. 17).

Regarding claim 58, [Splint] further discloses if the first value is a pointer, the second set of usability properties comprising the pointer pointing to a buffer with at least one readable element (see, e.g., p. 48 (MaxRead(ptr)>=0)).

Regarding claim 59, [Splint] further discloses if the first value is a scalar, the second set of usability properties comprises the scalar being initialized (see, e.g., section 3, beginning on p. 17).

Regarding claim 60, [Splint] further discloses if the first value is a struct, the second set of usability properties comprises each field of the struct being initialized (see, e.g., section 3, beginning on p. 17).

### ***Conclusion***

5. Any inquiry concerning this communication or earlier communications from the Examiner should be directed to Eric B. Kiss whose telephone number is (571) 272-3699. The Examiner can normally be reached on Tue. - Fri., 7:00 am - 4:30 pm. The Examiner can also be reached on alternate Mondays.

If attempts to reach the Examiner by telephone are unsuccessful, the Examiner's supervisor, Tuan Dam, can be reached on (571) 272-3695. The fax phone number for the organization where this application or proceeding is assigned is (571) 273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://pair-direct.uspto.gov>. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Eric B. Kiss/  
Eric B. Kiss  
Primary Examiner, Art Unit 2192